mg

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/685,272	10/09/2000	Eric M. Dowling	SEARCHP.008DV1	4446

27299 7590 05/23/2003

GAZDZINSKI & ASSOCIATES
11440 WEST BERNARDO COURT, SUITE 375
SAN DIEGO, CA 92127

EXAMINER	
VU, TUAN A	
ART UNIT	PAPER NUMBER
2124	6

DATE MAILED: 05/23/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 07-01)

P24

Office Action Summary	**Application No.** 09/685,272	**Applicant(s)** DOWLING, ERIC M.
	Examiner Tuan A Vu	**Art Unit** 2124

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 09 October 2000.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 24-29 and 33-76 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 24-29, 33-76 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on 09 October 2000 is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 2,4 .	6) ☐ Other: .

DETAILED ACTION

1. This action is responsive to the application filed October 9, 2000.

As per the pre-amendment filed 10/26/2000, claims 1-23, and 30-32 have been cancelled. Claims 24-29, and 33-76 are pending examination.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 24, 26-29, 33-34, 38-43, 47-52, 56-61,65-69, and 74-76 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rhim et al., USPN: 6,006,022 (hereinafter Rhim), in view of Baxter, USPN: 6,182,206 (hereinafter Baxter).

As per claim 24, Rhim discloses a method for programming a programmable system comprising a user programmable system with address interconnect unit (e.g. Fig. 16), the method comprising:

writing a first program in a first programming language, said first program configured to implement one or more user-defined address mapping function in said programmable address interconnect unit (e.g. col. 7, lines 1-3; col. 14, lines 21-37; Figs. 11, 16 – Note: configuration via hardware definition language is user-defined);

compiling said program; generating a first executable image, said image adapted for loading into a first program memory coupled to said system with address interconnect unit (e.g.

col. 8, lines 36-42 – Note: use of simulation tool with HDL inherently includes the compilation of such language in order to generate the hardware model);

writing a second program in a second programming language, such program configured to implement a desired algorithm (e.g. *high level language* - col. 12, lines 27-40);

compiling said second program into object code, said object code comprising a plurality of machine level instructions for a processor, such plurality of instructions comprising at least one instruction to control said system with address interconnect unit (e.g. *compile* - col. 12, lines 27-40 – Note: analyzing memory activities and machine states is equivalent to control activities of interconnect unit and memory/address operations and the implementation of some algorithm, or control or diagnostic instructions); and

generating a second executable image, said image adapted for loading into a second program memory coupled to said processor (see col. 12, lines 27-40 – Note: machine executable code loaded for execution is image of such second compiled program).

But Rhim does not specify that the programmable system with address interconnect unit from above is a programmable address arithmetic unit (PAAU); nor does Rhim specify that the user-defined address mapping functions are address calculation functions in said PAAU, and that the second program comprises instructions to control such PAAU. However, Rhim mentions the use of simulation/verification with an analyzer to perform load/store transactions analysis, and cache miss exception detection, and register instruction debug logic (e.g. col. 10, line 47 to col. 11, line 58); as well as system address data exception detection (e.g. col. 15, lines 19-35); and suggests emulating a embedded target processor and detect arithmetic operation malfunctions therein (e.g. col. 1, lines 23-27; col. 2, lines 13-19); and programmable memory to dynamically accommodate real-time changes in the build logic and interconnect of such processors (col. 5, lines 36-44). One skilled in the art would recognize the suggestion by Rhim to emulate embedded processor in order to dynamically capture faults in memory related operations, thus prevent resources usage for recovery from memory erroneous referencing or cache misses, hence to alleviate/optimize processor resources. Further, Baxter, in a method to dynamically reconfigure memory organization microprocessor using PLA to address hardware operation logic and internal changes analogous to the method of configure and emulate hardware activities and/or system logic as taught by Rhim, discloses the a re-programmable address operate unit (*AOU*) to perform address-related operations coupled with the instruction fetched by the emulated processor (e.g. col. 7, line 60 to col. 8, line 21; col. 28, lines 17-51; Figs. 10-11). Baxter, like Rhim, also suggests the emulation to address the architecture specificity and performance optimization of embedded system processors, like DSP (e.g. col. 1, lines 30-48). Hence, in view of the suggestion by Rhim and teachings by Baxter, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement a programmable address arithmetic unit as suggested by Baxter to Rhim's method of debugging the memory-related operations and address/register operations as taught by Rhim. One would be motivated to do so because keeping track of the dynamic machine state of the execution while emulating the hardware model in providing an programmable address arithmetic unit as claimed in order to addressing issues during processing the instructions (i.e. instructions of second program controlling the PAAU as claimed), and being able to reconfigure those addressing issues via such address arithmetic unit would alleviate subsequent memory faults in the iterative emulation process, and thereby optimizing the embedded microprocessor resources as have been suggested by Rhim and Baxter.

As per claim 26, Rhim discloses a hardware definition language (e.g. col. 14, lines 21-37).

As per claim 27, Rhim mentions assembler language associated with high-level language (e.g. col. 3, lines 55-65); but Baxter teaches assembly language (e.g. col. 39, lines 18-22). It would have been obvious for one of ordinary skill in the art at the time the invention was made to compile high-level language into a assembly language, in case Rhim's code translation of the second program has not been converted into assembly language, because such conversion as taught by Baxter would be applicable to the architecture of the target processor instructions set directly, hence enhance the platform-specific applicability.

As per claim 28, Rhim discloses that the second program is high level language (col. 12, lines 27-40).

As per claim 29, Rhim does not provide a special purpose circuitry for the programmable arithmetic unit; but discloses a design database (90 Design database – Fig. 1 – Note: this repository is equivalent to storing library for the first program to access design data) while Baxter discloses a circuitry to perform address operations (e.g. Fig. 10,11a-b) and further discloses the first program, a configuration logic language, to retrieve configuration data from a repository of architecture description (e.g. *memory 101, and reconfig. Logic 104, Access logic 102* – Fig. 5 – Note: configuration logic is equivalent to first program language as claimed). Hence, both Rhim/Baxter implicitly disclose the use of software libraries by the first program to access the PAAU circuitry as addressed above; thereby have rendered this instant claim obvious in view of the above teachings and also of the rationale as set forth in claim 1.

As per claim 33, Rhim discloses:

supplying first software module comprising instructions to implement an algorithm (e.g. col. 12, lines 27-40 – Note: analyzing memory activities and machine states is equivalent to control activities of interconnect unit and memory/address operations and the implementation of some algorithm, or control or diagnostic instructions);

supplying a second software module comprising configuration codes to define the operation of the user-defined address mapping function in the programmable interconnect circuitry (e.g. col. 7, lines 1-3; col. 14, lines 21-37; Figs. 11, 16 – Note: configuration via hardware definition language is user-defined), whereby

at least one instruction of said first module references an operand using such address mapping function in the interconnect circuitry (e.g. *load/store* – col. 10, line 60 col. 11, line 30 – Note: memory operations inherently include referencing of operands).

But Rhim fails to specify that the user-defined address mapping function is an user-defined addressing mode nor does Rhim specify that the interconnect circuitry is a programmable address arithmetic unit (PAAU). But the addressing mode operations limitation, i.e. user-defined addressing mode as claimed, and PAAU limitation have been addressed in claim 1 above using Baxter's teachings, hence would be rejected herein using the rationale set forth therein.

As per claim 34, Rhim suggests an embedded microprocessor (col. 1, lines 23-27) and Baxter discloses a emulating with HLL the operation of a DSP (col. 1, lines 30-48). One of ordinary skill in the art would have been motivated to apply the emulation of a embedded system as suggested by Rhim to an DSP as taught by Baxter, because DSP is handling computation-intensive or complex operations and that is exactly the reason for Rhim to emulate the operation of such complexity in order to capture dynamically the execution/exception state using an re-programmable interconnect system as disclosed, and thereby optimize its system resources utilization.

As per claim 38, in view of the combined teachings of Rhim and Baxter in claim 33 for the PAAU feature, this instant claim limitation would have been obvious because Rhim discloses configuring the PAAU blocks (e.g. PLA block *860* – Fig. 15) using the configuration code of said second program module.

As per claim 39, Rhim (with Baxter's teachings) only discloses sequence of input vectorized into sets, machine state snapshots, and register state observation per cycle (e.g. col. 11, lines 30-55; col. 12, lines 41-50; Fig. 5, 7) for emulation verification and testing but does not disclose micro-sequenced of state operations to implement the PAAU. Baxter discloses a instruction sequencer to enable facilitate instruction execution as a state machine (e.g. Instruction state sequencer 100 – Fig. 5; col. 18, line 63 to col. 19, line 16; Fig. 6). It would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the state observation sequences as suggested in the memory vector/snapshot/register state debug and verification techniques by Rhim to include the instruction state sequencer, i.e. micro-sequenced state machine, as taught by Baxter because this would facilitate the observation at micro-level state of the instruction as it evolved into the PAAU configured by the HDL or second module, and thereby enabling the dynamic adjustment of the address resolution scheme intended in designing and optimizing the PAAU as suggested by the combination Baxter/Rhim.

As per claim 40, Rhim discloses a matrix for interconnecting the elements of the FPD or PLA and a latch and debugging logic (e.g. element 801a, Fig. 2; Fig. 14-17) but does not specify a cross-bar switching element. Baxter discloses a cross-bar switch while configuring the programmable data operating units (e.g. Fig. 8). It would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the interconnect circuitry by Rhim so to include a cross-bar switch to handle the selective routing of inputs going into the data handling units. In view of the disclosed latch and debugging logic and the interconnect circuit suggested by Rhim, and the teachings to use a cross-bar switch to route the correct data into the unit under emulation, it would have been obvious for one of ordinary skill in the art at the time the invention was made to include the cross-bar switch as taught by Baxter to Rhim's method to latch and debug modules interconnected by the second program in order to handle the PAAU circuitry emulation and debug as addressed in claim 33 because this cross-bar would impart the selectivity of data entering the PAAU, thereby improve system debug error-preventing and efficiency.

As per claim 41, Rhim (combined with the PAAU teachings of Baxter) discloses dispatching subsets of instructions to functional units in a multi-use processor (col. 1, lines 13-60--Note: a microprocessor as suggested by Rhim is implicitly a multi-use processor), one of such functional units being the PAAU (e.g .col. 12, lines 20-62; Fig. 4).

As per claim 42, this is a computer-readable medium of claim 33 above, hence incorporates the rejection set forth therein. Further, this claim includes a computer-readable medium limitation, which Rhim/Baxter does not disclose. Official notice is taken that the use of a computer-readable medium to store a computer product program code was a well-known concept at the time of the invention. Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to use a computer-readable medium to store the computer product program code as disclosed by Rhim/Baxter because this would facilitate the distribution/sale of such computer product and the use of such product by a broader population of computer users.

As per claim 43, see corresponding rejection set forth in claim 34.

As per claims 47-50, these claims include the limitations of the corresponding claims 38-41, respectively; hence are rejected herein using the same grounds of rejection as set forth therein, respectively.

As per claim 51, this claim includes the limitations of claim 42 such as comprising

a first software module,

a second software module, and the steps of

executing (first module),

using (configuration codes), and

referencing an operand (using addressing mode); hence incorporates the corresponding rejections as set forth accordingly in claim 42. Further, this claim includes the PAAU limitation and the user-defined addressing mode supplied by the PAAU configured by the second module. This PAAU/the addressing mode limitation has been addressed in claim 33 above; hence is rejected herein using the same rationale set forth in claim 33.

As per claim 52, see corresponding rejection set forth in claim 34.

As per claims 56-59, these claims include the limitations of the corresponding claims 47-49, respectively; hence are rejected herein using the same grounds of rejection as set forth therein, respectively.

As per claim 60, Rhim discloses a system adapted for loading a first software module having a plurality of instructions and a second software module having a configuration code, the system comprising: a processor having a programmable interconnect unit (e.g. Fig. 16); such processor adapted to

execute one first instruction to implement an algorithm (e.g. col. 12, lines 27-40);

configure an user-defined address mapping function of the programmable interconnect circuitry (e.g. col. 7, lines 1-3; col. 14, lines 21-37; Figs. 11, 16 – Note: configuration via hardware definition language is user-defined);

execute one second instructions referencing an operand using said user-defined address mapping function in the programmable interconnect circuitry (e.g. *load/store* – col. 10, line 60 col. 11, line 30 – Note: memory operations inherently include referencing of operands).

But Rhim fails to specify that the user-defined address mapping function of the interconnect circuitry is an user-defined addressing mode nor does Rhim specify that the programmable interconnect unit is a programmable address arithmetic unit (PAAU). But the addressing mode operations limitation, i.e. user-defined addressing mode as claimed, and PAAU limitation have been addressed in claim 1 above using Baxter's teachings, hence would be rejected herein using the rationale set forth therein.

As per claim 61, see rejection of claim 34.

As per claims 65-68, these claims include the limitations of the corresponding claims 38-41, respectively; hence are rejected herein using the same grounds of rejection as set forth therein, respectively.

As per claim 69, Rhim discloses a computer-implemented method for programming a processor comprising a programmable interconnect unit, the method comprising:

allowing a sequence of instructions for implementing an algorithm to generate memory operations (col. 12, lines 27-40);

observing said sequence of memory operations (e.g. col. 10, line 47 to col. 11, line 58; col. 15, lines 19-35); and

generating a configuration program for said interconnect unit, said configuration program defining a user-defined address mapping function (e.g. col. 7, lines 1-3; col. 14, lines 21-37; Figs. 11, 16).

But Rhim does not specify that the programmable interconnect unit is a programmable address arithmetic unit, nor does Rhim specify that the address mapping function defined by the configuration program is an user-defined addressing mode. But in view of the teachings by Baxter as mentioned in claim 24 combined with the suggestion by Rhim, these limitations have been addressed in the corresponding rejection set forth in claim 24.

Nor does Rhim specify that generating memory operations from executing the algorithm is generating a sequence of addresses; and observing such memory operations is observing a subsequence of said sequence of addresses. In view of the teachings by Baxter for having an address calculating unit (e.g. col. 7, line 60 to col. 8, line 21; col. 28, lines 17-51; Figs. 10-11 – Note: the following of group of address inputs for implementing the arithmetic operations therefrom is equivalent to observing a subsequence of address data going into the address operate unit), the above limitations would have been obvious because of the rejection set forth in claim 1 and also because generating and observing the subsequence of address arithmetic operations are implicitly implied in the teachings of Baxter.

Nor does Rhim disclose that the user-defined address operation is capable of regenerating the subsequence of addresses as mentioned above. This subsequence of addresses limitation has been addressed above in view of Baxter's teachings combined with Rhim's; and the regenerating of such subsequence would have been implicitly disclosed by both Rhim and Baxter because the re-adjusting capability associated with the re-programmability or re-configurability of both systems disclosed by Rhim or Baxter, i.e. any time a address generating functions encounters an error, the user-defined programmable configuration program or the implementation of the algorithm would be corrected to remedy to the problem and regenerate an improved subsequence of address arithmetic operations (Rhim: *easily reconfigurable* - col. 5, line 56 to col. 6, line 25).

As per claim 74, Rhim discloses a method of executing software in a computerized system, such system comprising processing data with programmable means for addressing (e.g. Fig. 14,16), a first software module containing instructions defining operations of an algorithm (e.g. col. 12, lines 27-40), and a second software module containing configuration codes defining the operation of an user-defined address mapping using means for addressing (e.g. col. 7, lines 1-3; col. 14, lines 21-37; Figs. 11, 16), the method comprising:

executing,

using, and

referencing an operand just as claimed in method claim 51.

The steps of executing, using, referencing would be rejected using the same grounds used in claim 51 above.

However Rhim fails to disclose that the configuration codes define user-defined addressing mode supplied by means of addressing; nor does Rhim disclose configuring the operation of such addressing mode to reference an operand being accomplished in part with said user-defined addressing mode. But the addressing mode operations limitation, i.e. user-defined addressing mode as claimed, as well as the address arithmetic operation have been addressed in claim 1 above using Baxter's teachings, hence would be rejected herein using the rationale set forth therein.

As per claim 75, Rhim discloses a computerized system adapted for loading a first software module having instruction means (e.g. col. 12, lines 27-40) and a second software module having one configuration code means (e.g. col. 7, lines 1-3; col. 14, lines 21-37), the system comprising:

means for processing data having programmable memory addressing means (e.g. Figs. 11,14, 16 – Note: interconnect circuitry of processor is equivalent to means to implement addressing of data communicating with memory);

whereby said means for processing is adapted to

execute at least one first instruction means for implementing an algorithm (e.g. *high level language* - col. 12, lines 27-40);

configure an user-defined memory operation in said programmable memory addressing means using said configuration (col. 7, lines 1-3; col. 14, lines 21-37; Figs. 11, 16); and execute at least one second instruction means, said second instruction means referencing an operand using said addressing means (e.g. *load/store* – col. 10, line 60 col. 11, line 30).

But Rhim does not specify that the configured user-defined memory operation is an user-defined addressing mode in the programmable memory addressing means, nor does Rhim specify referencing an operand using such user-defined addressing mode. But the addressing mode limitation, i.e. user-defined addressing mode as claimed, along with the programmable addressing means for address arithmetic have been addressed in claim 1 above using Baxter's teachings, hence would be rejected herein using the rationale set forth therein.

As per claim 76, Rhim discloses a computerized system, comprising

means for implementing (algorithm ..),

means for defining (an user-defined …), and

whereby implementing the algorithm references (an operand …)

such as recited in claim 75.

Hence those above means limitations are rejected with the corresponding rejections used in claim 75, including the features not disclosed by Rhim which have been addressed for obviousness in view of Baxter as set forth therein.

4. Claims 25, 35-37, 44-46, 53-55, 62-64, and 70-73 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rhim et al., USPN: 6,006,022, and Baxter, USPN: 6,182,206, as applied to claims 1, 33, 42, 51, 60, 69 above, and further in view of Stan Liao et al., "*Storage Assignment to Decrease Code Size*", June 1995, In Proceedings of ACM SIGPLAN'95, Conference on Programming Language Design and Implementation, Vol. 30, Issue 6, pp. 235—253 (hereinafter Liao).

As per claim 25, Rhim does not disclose the address calculation function is invoked by an auto-addressing mode. Nor does Baxter specify that the address operate unit has address calculation function invokable by an auto-update addressing mode. But both Baxter and Rhim suggest a method for build a programmable an embedded processor (e.g. Baxter: DSP- col. 1, lines 30-48) circuitry with the memory, bus, and addressing functionalities and executing those functionalities using a high-level language to implement an algorithm, e.g. implementing the memory operations or addressing operations as above-mentioned in claim 1. Liao, in an optimization method using high-level programming code to implement addressing modes of embedded systems analogous to the address operate unit implementation/emulation by Baxter, discloses direct addressing modes with auto-increment/decrement arithmetic in order to enhance the limited addressing capabilities of embedded signal processors (e.g. pg. 2, last 3 paragraphs and pg. 3, 1st paragraph). It would have been obvious for one of ordinary skill in the art at the time the invention was made to enhance the address arithmetic unit as mentioned in the combined teachings/suggestions by Rhim/Baxter as in claim 1 above so to provide a functionality operable to perform with the arithmetic operations for auto-increment/decrement as taught by Liao, i.e. auto-update addressing mode as claimed because this would further accommodate and enhance for the limited addressing capabilities, the direct addressing modes, of the resource-limited and special instruction architecture of the embedded microprocessors as suggested by Rhim/Baxter.

As per claim 35, the combination Rhim/Baxter fails to disclose an instruction causing an auto-update, such auto-update operation being part of the addressing mode defined by the programmable system by the user. But Liao has disclosed the auto-update limitation as from above; and moreover, Liao discloses an operation with pointer operand that causes such auto-update operation (e.g. pg. 6, assembly code of Fig. 2 – Note: any indexed number is equivalent to a operand pointer causing an auto-increment/decrement, or offset assignment). It would have been obvious for one of ordinary skill in the art at the time the invention was made to enhance the address arithmetic unit as mentioned in the combined teachings/suggestions by Rhim/Baxter as in claim 33 above so to provide a functionality operable to perform with the arithmetic operations for auto-increment/decrement as taught by Liao, i.e. auto-update applying to operand pointer in addressing modes, because of the same reasons as set forth in claim 25 above.

As per claim 36, Liao also discloses assembly language mnemonic (see claim 35 – pg. 5, col. c -- Fig. 1).

As per claim 37, the combination Rhim fails to disclose an instruction used to specify one of the user-defined addressing modes to be selected to define the operation of auto-update. However, Baxter discloses a instruction set architecture or ISA instruction, i.e. the first module code as claimed, wherein is defined specific addressing modes (e.g. col. 12, lines 40-50) whereas Liao, in the high-level language or HLL-implemented algorithm to address instruction in a embedded architecture as mentioned in claim 25, discloses code generation step that selects addressing modes, i.e. *single indexing register with indirect auto-increment, and auto-decrement addressing modes* (pg. 3, first 3 paragraphs). It would have been obvious for one of ordinary skill in the art at the time the invention was made to modify Rhim's method to execute the configurable memory operations and code implemented circuitry thus disclosed so to include the addressing-mode-selecting code instruction as suggested by both Baxter and Liao, because this instruction to select the appropriate addressing mode in order to define which auto-update operation to implement would further enhance the architecture specificity as mentioned by Baxter, and thereby would alleviate resources usage for handling each addressing scenario given the limited register, storage and error handling capacities as known to be associated with the embedded systems like DSP as suggested by Baxter/Rhim and by Liao when Liao attempts to reduce the implementing code to cover the addressing operations.

As per claims 44-46, these claims include the limitations of the corresponding claims 35-37, respectively; hence are rejected herein using the same grounds of rejection as set forth therein, respectively.

As per claims 53-55, these claims include the limitations of the corresponding claims 35-37, respectively; hence are rejected herein using the same grounds of rejection as set forth therein, respectively.

As per claims 62-64, these claims include the limitations of the corresponding claims 35-37, respectively; hence are rejected herein using the same grounds of rejection as set forth therein, respectively.

As per claim 70, Rhim discloses event detection and monitoring of vector, registry operations (e.g. col. 10, line 60 to col. 11, 55) whereas Baxter discloses an instruction state sequencer (Instruction state sequencer 100 – Fig. 5; col. 18, line 63 to col. 19, line 16; Fig. 6, 17B), hence both suggest a monitoring of a history of a state of an instruction, but neither Baxter nor Rhim specify observing an address history sequence of pointer variable. In view of the Liao's teaching using analysis of assembly code sequence including a pointer operand (pg. 6, assembly code of Fig. 2 – Note: any indexed number is equivalent to a operand pointer causing an address auto-increment/decrement, or offset assignment) combined with the suggestion by

Rhim/Baxter, it would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the monitoring/observation suggested by both Rhim/Baxter of the memory-related operations, e.g. address arithmetic sequence observation, so that the observation corresponds to the history sequence of pointer operand, i.e. pointer variable as claimed, as suggested by Liao, because this would enhance the debugging of address arithmetic operations in that previous operands or variables and subsequent operations resulting therefrom would be recorded allowing immediate remedy be taken should any memory operation conflicting occurs.

As per claim 71, only Baxter suggest address generating and rerouting of address inputs (e.g. col. 7, line 60 to col. 8, line 21; Figs. 10-11) but does not specify defining an auto-update operation nor advancing from a current address element of a subsequence to a successive address-element of such subsequence. But in view of Liao's teaching to auto-increment or decrement of address element while observing a subsequence of assembly language code instructions (pg. 5, Fig. 1) and a necessity to subsume auto-update address operations in DSP to remedy for the lack therein for index with offset addressing mode(e.g. pg. 2, last paragraph to pg. 3, first paragraph), , it would have been obvious for one of ordinary skill in the art at the time the invention was made to include in the monitoring and debugging operations suggested by Rhim/Baxter the auto-increment techniques by Liao with the advancing of address-element into successive address-element as claimed, because this would help overcome deficiency in the instruction architecture of such embedded processors, i.e. DSP, as suggested by Liao, in the context that both Baxter and Rhim's methods emulate an embedded processor of the likes of DSP (see claim 34).

As per claim 72, this limitation is analogous to the limitations recited in claims 35, and 36, hence incorporates herein the corresponding rejection set forth therein; and further includes the limitation as to the pointer undergoing a subsequence address change. But this limitation has been addressed in claim 71 above.

As per claim 73, neither Baxter nor Rhim specify reducing the N cycles of instructions to M cycles of instructions when executing the auto-update of claim 72. But in view of Liao's teachings as mentioned in claim 71 in terms of successively auto-incrementing or decrementing address pointer variables to overcome the lack of addressing with offset in a DSP, the reducing of N cycles into M cycles limitation would have been obvious by virtue of the rejection set forth in claim 71.

Conclusion

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Pat No. 5,883,814 to Luk et al., disclosing HDL configuration and execution for test of DRAM/microprocessor.

U.S. Pat No. 5,247,627 to Murakami et al., disclosing DMA/arithmetic unit on DSP and memory data control/transfer.

Sudarsanam et al., "Analysis and Evaluation of Address Arithmetic Capabilities in Custom DSP Architectures", June 1997, Design Automation Conference, pp. 287-292.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 746-7239, (for formal communications intended for entry)

or: (703) 746-7240 (for informal or draft communications, please label "PROPOSED" or "DRAFT")

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

[illegible]

VAT
May 19, 2003

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100